VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

INTEGRACE SERVERU UNDERTOW SE SYSTÉMEM JENKINS CI

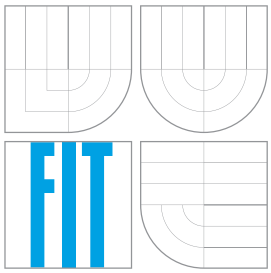
SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE
AUTHOR

Bc. JAKUB BARTEČEK

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

INTEGRACE SERVERU UNDERTOW SE SYSTÉMEM JENKINS CI

INTEGRATION OF JENKINS CI WITH UNDERTOW

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE
AUTHOR

Bc. JAKUB BARTEČEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PETR MÜLLER

BRNO 2014

Abstrakt

Tento semestrální projekt se zabývá nahrazením webového serveru v systému Jenkins CI za server Undertow. Server Undertow by měl být potenciálně rychlejší než současné řešení a celkově lepší. V práci jsou popsány obecné informace o komponentách a programech, které se této problematiky týkají. Nejdůležitější částí práce je analýza současného stavu a návrh způsobu integrace. Samotná integrace není provedena v rámci semestrálního projektu, ale je předmětem navazujícího diplomového projektu.

Abstract

This term project deals with replacement of webserver in Jenkins CI by server Undertow. The Undertow should be potentially faster than actual state and in general better. In this project there are described general information about components and programs, which are related to this topic. The most important part is analysis of current state and a design of integration. The integration is not accomplished in this term project. Is is a subject of follow-up diploma thesis.

Klíčová slova

Jenkins, Undertow, servlet, integrace, kontinuální integrace, Winstone, Jetty, Java

Keywords

Jenkins, Undertow, servlet, integration, continuous integration, Winstone, Jetty, Java

Citace

Jakub Barteček: Integrace serveru Undertow se systémem Jenkins CI, semestrální projekt, Brno, FIT VUT v Brně, 2014

Integrace serveru Undertow se systémem Jenkins CI

Prohlášení

Prohlašuji, že jsem tento semestrální projekt vypracoval samostatně pod vedením pana inženýra Petra Müllera. Další informace mi poskytl pan doktor Vojtěch Juránek, který je zaměstnancem firmy Red Hat®.

.....
Jakub Barteček
11. ledna 2014

Poděkování

Děkuji panu inženýru Petru Müllerovi za vedení mého semestrálního projektu a panu doktoru Vojtěchu Juránkovi za odborné konzultace týkající se zpravovávané problematiky.

Obsah

Kapitola 1

Úvod

Tento semestrální projekt se zabývá vylepšením serveru Jenkins CI, který je v praxi velmi využíván pro potřeby průběžného testování softwaru a jeho kontinuální integraci. Vylepšení se týká především webového serveru a *servlet* kontejneru, který je v Jenkins CI integrován. V současném stavu tyto funkce vykonává kombinace nástrojů Winstone a Jetty.

Server Winstone je již neudržovaný a zastaralý nástroj a z tohoto důvodu byl z velké části nahrazen serverem Jetty, který potřebnou funkcionalitu poskytuje. Server Jetty je poměrně komplexní projekt a nabízí mnoho funkcionality, ale na druhou stranu jeho rozsah nedovoluje poskytovat maximální rychlost.

V současné době vznikl nový webový server Undertow, který si klade za cíl být co nejjednodušší a nejrychlejší a mohl by být přínosný a vhodný pro Jenkins CI. Jelikož tento server je nový a je sponzorován firmou Red Hat, tak lze předpokládat, že jeho vývoj bude nadále pokračovat a nebude zastarávat.

Cílem této práce je nahradit server Jetty a případně i serveu Winstone pomocí serveru Undertow a integrovat jej se serverem Jenkins CI. Při integraci je kladen důraz na snahu zachovat zpětnou kompatibilitu se starým řešením.

V rámci tohoto semestrálního projektu jsou nejprve rozebrány potřebné informace týkající se jednotlivých nástrojů a plánované integrace. Následně je detailně analyzována architektura Jenkins CI a způsob jeho integrace se servery Winstone a Jetty. V následující části je diskutována varianta nahrazení pouze serveru Jetty a varianta nahrazení serveru Jetty i Winstone pomocí serveru Undertow. Z provedené analýzy je zvolena jedna varianta, která je vybrána pro následnou integraci.

Samotná implementace integrace není předmětem tohoto semestrálního projektu a bude provedena až v navazujícím diplomovém projektu. V rámci dimplomového projektu bude také provedeno testování výkonu modifikovaného systému Jenkins CI a celkové shodnocení navržených a provedených změn.

Kapitola 2

Jenkins CI a související nástroje

Tato kapitola se zaměřuje na teoretické základy práce, které je nutné nebo vhodné znát pro pochopení zpracovávané problematiky. Je zde detailněji popsán systém Jenkins CI (kapitola 2.1) ke kterému se tato práce přímo váže. S ním je spojeno seznámení s webovými servery Winstone (kapitola 2.4) a Jetty (kapitola 2.5), jejichž kombinace je současně v Jenkins CI integrována. Pro tuto práci je důležité poruzumět rozdílu mezí *servlet kontejnerem* a webovým server. Vysvětlením těchto pojmů se zabývá kapitola 2.2.

Větší důraz je dále věnován serveru Undertow (kapitola 2.6), který byl vybrán jako nový webový server pro Jenkins CI. V tomto případě je provedena hlubší studie tohoto nástroje, aby na jejím základě bylo možné pochopit a provést samotnou integraci s Jenkins CI, která je jádrem této práce. Pro poskytnutí ucelených informací je menší prostor také vymezen pro nástroj Maven (kapitola 2.3), který se používá k řízení překladu serveru Jenkins CI, a je v této práci několikrát zmiňován.

Uvedené informace mají spíše informativní charakter, aby poskytly ucelený úvod do zkoumané problematiky. Jsou zaměřeny především na informace týkající se samotné integrace. Pro případné získání detailnějších informací jsou uvedeny patřičné zdroje, kde je lze hledat.

2.1 Jenkins CI

Jenkins CI je komunitní open source nástroj pro kontinuální integraci softwaru, který je vyvíjen pod svobodnou licencí MIT[1] [2]. Je velmi populární a využíván malými i velkými firmami jako je například firma Red Hat, kde tento program běží na stovkách serverů. Původní název tohoto projektu je Hudson[2]. Když se vývoje tohoto projektu ujala firma Oracle, tak se tento projekt rozštěpil a vznikla komunitní verze projektu, což je server Jenkins CI. Přesto se v některých částech tohoto projektu stále objevuje název Hudson, ale jedná se pouze pozůstatek z původního projektu.

Zkratka CI je z anglického spojení *continuous integration*, což lze do češtiny přeložit jako kontinuální nebo průběžná integrace. Krátké seznámení se z touto metodologií je v následující kapitole.

Informace v této kapitole byly především čerpány z knihy [4], kde lze nalézt další informace o serveru Jenkins CI, a také z webové stránky projektu [6].

[1] Licence MIT: http://opensource.org/licenses/MIT

[2] Webové stránky projektu Hudson: http://hudson-ci.org/

2.1.1 Kontinuální integrace a využití Jenkins CI

V minulosti byla integrace programu do výsledného produktu velmi náročným procesem a často ztraceným časem. S vydáním každé verze programu se musel postup probíhající před vydáním produktu opakovat a pro vývojářský tým to prakticky znamenalo zdržení se. Pokud se v tomto procesu odhalil nějaký problém, což bylo běžné, tak jeho řešení bylo z důvodu nedostatku času a jeho pozdního objevení mnohem problematičtější, než kdyby byl tento problém odhalen dříve.

Kontinuální integrace je moderní přístup k vývoji softwaru, který mění způsob myšlení nad celým procesem vývoje a snaží se předcházet problémům popsaným výše a především ušetřit čas. V tomto přístupu je využíván nějaký kvalitní nástroj, který automatizovaně provádí specifikované kroky, které provázejí integraci softwaru a jeho vydání.

Jedním z nástrojů poskytujících podporu pro kontinuální integraci při vývoji softwaru je server Jenkins CI.

Základními možnostmi, které umožňuje Jenkins CI nakonfigurovat, jsou:

- Spouštění integračních a jednotkových testů v přesně definovaném čase (např. v noci, kdy jsou servery méně vytížené)
- Spuštění integračních a jednotkových testů při změně ve verzovacím systému. Jenkins CI dokáže zaznamenat změnu v repozitáři, stáhnout si změny a spustit testování
- Shromažďování a vyhodnocování metrik vývoje softwaru
- Spuštění akceptačních testů
- Informování e-mailem o testech, které skončily chybou
- Automatické nahrání nové verze produktu na server

Uživatelé mohou kdykoliv přidat využití libovolné funkcionality systému a tudíž neopakovat stále stejné kroky.

2.1.2 Způsoby použití aplikace

Celý program je napsán v jazyce Java a je tedy plně přenositelný mezi platformami. Architektura je navržena tak, aby byla lehce rozšiřitelná pomocí tzv. *pluginů*, kterých je pro Jenkins vytvořené velké množství.

Jenkins CI je určen pro běh na serveru a je dostupný přes webové rozhraní (ale může být samozřejmě spuštěn na libovolném osobním počátači). Komunikuje pomocí protokolu *HTTP*, který je založen na modelu *požadavek-odpověď* (anglicky *request-response*). Jelikož tento způsob komunikace je velmi běžný, tak není v programu přímo implementován, ale využívá k němu externí nástroje. Dalším nástrojem, který pro svou činnost Jenkins potřebuje, je servlet kontejner ve kterém samotná aplikace poběží. Tento pojem je blíže objasněn v kapitole 2.2.

Existují dvě možnosti jak spustit server Jenkins CI:

1. Může běžet na libovolném *Java EE* aplikačním serveru [1] jako jsou například servery JBoss[3] anebo Glasfish[4]. Jenkins CI se standardně nasadí na server (dle zvyklostí

[3] Více informací o serveru viz http://www.jboss.org/jbossas/
[4] Více informací o serveru viz http://www.oracle.com/technetwork/middleware/glassfish/

konkrétního serveru) a poté je s ním možné pracovat. V tomto případě veškerou nízkoúrovňovou komunikaci pomocí *HTTP* i práci servlet kontejneru zajišťuje aplikační server.

2. Pokud nechceme nebo nemůžeme spouštět Jenkins CI na aplikačním serveru, tak jej lze spustit přímo z vytvořené `.war` archivu (překladem se zabývá kapitola 2.1.3). V tomto případě se o práci servlet kontejneru i webového serveru komunikujícího protokolem *HTTP* stará kombinace nástrojů Winstone a Jetty, které jsou přímo integrovány do serveru Jenkins CI.

   Nahrazení těchto dvou nástrojů (nebo pouze serveru Jetty) a zajištění vykonávání této činnosti je hlavním cílem této práce. Záměrem je tedy nahradit server Jetty a případně i server Winstone pomocí zvoleného nového serveru Undertow. Studie těchto nástrojů a jejich rozbor je předmětem následujících kapitol.

Pro tuto práci je důležitý druhý způsob používání Jenkins CI a proto další informace se budou přímo vázat k němu.

2.1.3 Základy práce s Jenkins CI

Pro přeložení a spuštění aplikace je potřeba pracovat z příkazové řádky nebo provést instalaci nějakým dávkovým souborem (skriptem). Aplikaci je možné stáhnout připravenou přímo ze stránek projektu [6], ale pro tuto práci je potřeba pracovat s aplikací ze zdrojových souborů. Aktuální verze aplikace je dostupná na serveru GitHub[5].

Po stažení zdrojových souborů je potřeba provést překlad aplikace. Pro tento automatizovaný překlad se používá nástroj Maven, který krátce zmíněn v kapitole 2.3. Pro provedení překladu bez spouštění jednotkových testů je potřeba použít tento příkaz:

```
mvn clean install -pl war -am -DskipTests
```

Pokud bychom chtěli spustit jednotkové testy aplikace, tak jednoduše vynecháme poslední přepínač. Nejjednodušším způsobem kontroly provedených změn v programu je právě spouštění automatizovaných testů a proto je tato možnost velmi důležitá pro plánovanou integraci se serverem Underotw. Kromě jednotkových testů obsahuje Jenkins CI také integrační testy, které jdou více do hloubky aplikační logiky a jsou schopny odhalit větší množství chyb, ale na druhou stranu jejich vykonání trvá dlouhou dobu (řádově více než hodinu). Integrační testy lze spustit příkazem:

```
mvn clean package
```

Další nevýhodou integračních testů je, že je běžný stav, kdy některé testy, aplikace která je překládaná z aktuálních zdrojových souborů, končí s chybou. Proto je při kontrole nutné porovnávat výsledky testů před provedením změn a po jejich provedením. Tento způsob kontroly aplikace bude využit při hodnocení navazující diplomové práce.

Po úspěšnm překladu aplikace vznikne v adresáři `./war/target/` archiv `jenkins.war`, který obsahuje celou přeloženou webovou aplikaci včetně nástrojů na kterých závisí. Z tohoto archivu je možné aplikaci přímo spustit pomocí příkazu:

[5] Adresa aktuální verze Jekins CI: www.github.com/jenkinsci

```
java -jar jenkins.war
```

Je možné nastavit ještě další parametry při spuštění programu, které lze vypsat pomocí přidání parametru `--help`.

Pro práci se spuštěnou instancí aplikace se používá především webové rozhraní. Standardně aplikace komunikuje pomocí protokolu *TCP* na portu 8080. Je tedy možné se na lokálním počítači připojit do aplikace zadáním URL adresy `http://localhost:8080` do webového prohlížeče.

Pokud se aplikaci povede spustit, tak je již možné libovolně pracovat pouze s prohlížeče. Detaily práce s aplikací Jenkins CI lze nalézt v této knize [4], ale tyto informace jsou již nad rámec této publikace.

2.2 Webový server a servlet kontejner

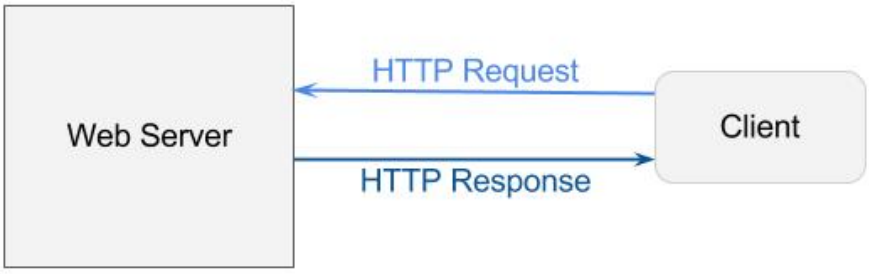
V této kapitole jsou vysvětleny pojmy webový server a servlet kontejner, které se na mnoha místech této práce objevují. Porozumění těmto pojmům je důležité, protože bez jejich znalosti by některé kapitoly byly těžko pochopitelné.

Informace zde uvedené byly čerpány z článku [5].

2.2.1 Webový server

Webový server je program, který zprostředkovává komunikaci přes síť s klienty, kteří se k němu připojí a požadují po něm nějaká data. Tato komunikace probíhá pomocí protokolu *HTTP*, který je založen na modelu *požadavek-odpověď* (anglicky *request-response*). Model této komunikace je zachycen na obrázku 2.1.

Typický způsob komunikace webového serveru je, že klient pomocí URL adresy specifikuje požadavek na nějaká data a server v odpovědi tato data pošle. Pokud by neexistoval za webovým serverem nějaký další program, tak by webový server vždy odpovídal na stejný požadavek stále stejnou odpovědí.



Obrázek 2.1: Model komunikace webového serveru s klientem [5]

2.2.2 Servlet kontejner

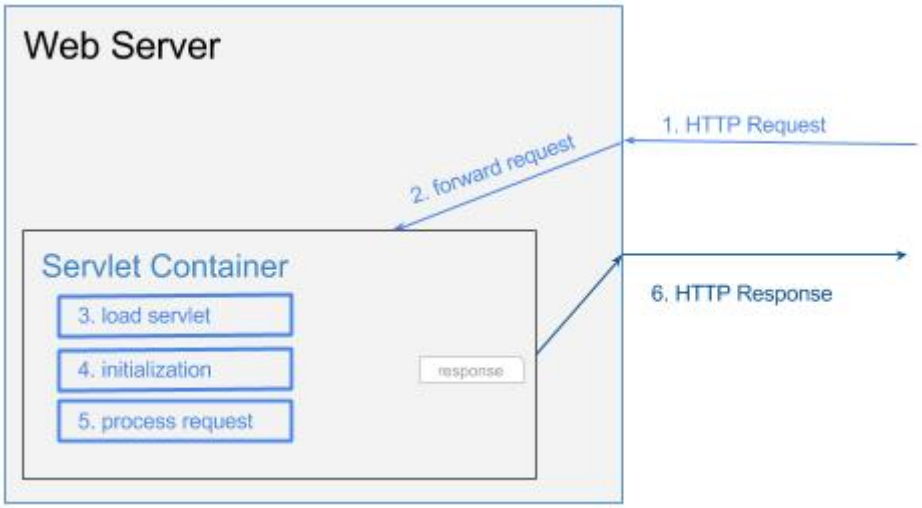
Jelikož dostávat stále stejná statická data při stejném požadavku není dostatečná funkcionalita serveru, tak existují způsoby jak zajistit dynamickou práci s daty na serveru (a tudíž

i měnící se odpovědi na stejný dotaz). Jedním ze způsobů jak této činnosti docílit je využití tzv. *servlet kontejneru* a *servletů*, které jsou navrženy pro programovací jazyk Java.

Servlet je standardní program (konkrétně jedna třída) napsaný v jazyce Java, který implementuje rozhraní `javax.servlet`. Implementace tohoto rozhraní ho zavazuje k definování několika metod, ale jinak se jedná o běžnou třídu z které jsou při zpracovávání požadavků vytvářeny její instance.

Základní myšlenkou servlet kontejneru je umožnit dynamicky vytvářet odpovědi (často webové stránky) pomocí vykonávání servletů. Samotný servlet kontejner je program, který poskytuje běhové prostředí pro vykonávání servletů, zajišťuje jejich vytváření, vykonávání a odstraňování. Dále se také podílí na zpracovávání HTTP požadavků, které jsou mu předávány od webového serveru.

Popsaný způsob fungování servlet kontejneru je znázorněn na obrázku 2.2.



Obrázek 2.2: Ukázka způsobu činnosti servlet kontejneru a jeho spolupráce s webovým serverem [5]

2.3 Maven

Základní obecné informace, něco o pomkách - co tam najdeme, způsobech překladů [9]

2.4 Winstone

Winstone je servlet kontejner, který je velmi jednoduchý a poskytuje funkcionalitu servlet kontejneru aniž by byl zatížen velkým množstvím požadavků, které jsou ve specifikaci jazyka Java EE. Nikdy neposkytoval veškeré služby servlet kontejneru, které specifikace jazyka definuje. V jeho názvu je zahrnutý pouze pojem servlet kontejner, ale tato aplikace vykonává i služby webového serveru odpovídající popisu v kapitole 2.2.

Hlavními cíli projektu bylo poskytovat funkcionalitu servlet kontejneru pouze pro jednu aplikaci, což je opačný přístup než u běžných aplikačních serverů jako jsou Glasfish, JBoss,

aj. Díky jeho omezeným službám je jeho velikost velmi malá a umožňuje jednoduchou integraci s cílovou aplikací.

Uvedené informace byly čerpány z oficiální stránky projektu Winstone [3].

2.4.1 Nedostatky servlet kontejneru Winstone

Původní myšlenka jednoduchého servlet kontejneru byla pro projekt Jenkins CI zajímavá. Kontejner Winstone má několik zásadních nedostatků kvůli kterým bylo časem jeho využití v Jenkins CI problematické [**?**].

Vývoj projektu Winstone již před delší dobou ustal a tím pádem nebyla poskytována žádná další podpora pro řešení a opravování objevených nedostatků a chyb. Značné množství bezpečnostních chyb objevených v projektu Jenkins CI bylo právě způsobeno tímto servlet kontejnerem.

Zpočátku možnosti servlet kontejneru postačovaly, ale časem je potřeba, aby byly přidávány nové funkcionality, které odpovídají aktuálním trendům a novým specifikacím. Příkladem mohou být nové specifikace servlet kontejneru, vývoj nových technologií jako webové sokety[6](angl. *websocket*).

2.4.2 Vývoj v komunitě Jenkins CI

Po ukončení vývoje projektu Winstone se musela o potřebné úpravy starat komunita Jenkins CI. Takováto práce je pro komunitu velmi zatěžující a naprosto neefektivní. Byly prováděny především nutné opravy bezpečnostních chyb v kontejneru, ale jinak aplikace dále degenerovala. Pro tento vývoj vznikl v projektu Jenkins CI nový repozitář[7] ze kterého jsou uváděné informace také čerpány.

Vývoj tímto způsobem probíhal až do verze *0.9.10-jenkins-47*. Když byl projekt v této fázi, tak vznikalo zadání tohoto diplomového projektu, které mělo za cíl výrazně zlepšit aktuální stav servlet kontejneru v projektu Jenkins CI. Během formulace zadání byl servlet kontejner v projektu přepracován a část byla nahrazena serverem Jetty, který je popisován v kapitole 2.5. Tímto krokem vznikla interní verze kontejneru *Winstone 2.0*. Tato změna proběhla velmi narychlo a nebyla detailně otestována. Obsahuje ještě množství nepotřebného kódu a samotná implementace není moc předhledná [**?**].

2.5 Jetty

[8] obecné seznámení, co to umí

2.6 Undertow

Obecné informace, k čemu je dobrý, srovnání s Jetty, odkazy na výsledky testování výkonu

Jak se používá, jak se s ním pracuje a programuje, detailněji argumenty v pdfku k wildfly8 - bezpecnost, rychlost, ...

[6] Oficiální stránky specifikace webových soketů http://www.websocket.org/

[7] Repozitář, kde komunita Jenkins CI provádí úpravy projektu Winstone: https://github.com/jenkinsci/winstone

2.6.1 Srovnání s Jetty

[7]

Kapitola 3

Analýza současného stavu a návrh integrace

3.1 Aktuální stav architektury servlet kontejneru v Jenkins CI

Popis aktuálního stavu architektury Jenkins z pohledu servlet kontejneru,

možnosti integrace: napojení Winstone a Jetty v Jenkins extras-executebles-war, vkládání winstonu jako .jar

3.2 Zjištěné problémy

Rozdílná verze Javy - Undertow v. 7, Jenkins v. 6 =¿ nutnost převést Jenkins na verzi 7, což mírně omezuje okamžité využití v hlavní větvi, ale do budoucna určitě bude nějaký přechod

3.3 Zpětná kompatibilita

zachování funkčnosti parametrů - závisí na tom různé komponenty a pluginy Jenkins. Odcitovat Keysukeho článek na docs.gmail.

Možnost řízení security věcí komunikace přes HTTP komunikace přes HTTPS AJP13 SPDY - aktuálně není v Jenkins využíván, pouze je podporován Jetty

3.4 Varianta ponechání Winstonu

3.5 Varianta nahrazení Jetty i Winstonu

3.6 Zvolená varianta

Způsob vyhodnocení správnosti výsledného řešení pomocí testů jenkins

Kapitola 4

Závěr

Literatura

[1] Kawaguchi, K.: Containers – Jenkins – Jenkins Wiki. `https://wiki.jenkins-ci.org/display/JENKINS/Containers` , 2011-02-03 [cit. 2014-01-06].

[2] Kawaguchi, K.: Governanace Document – Jenkins – Jenkins Wiki. `https://wiki.jenkins-ci.org/display/JENKINS/Governance+Document` , 2012-03-21 [cit. 2014-01-09].

[3] Knowles, R.: Winstone servlet container [online]. `http://winstone.sourceforge.net/` , cit. 2014-01-09.

[4] Smart, J. F.: *Jenkins: The Definite Guide*. O'Reily Media, Inc., 2011, ISBN 978-1-449-30535-2.

[5] Wang, R.: What is a Servlet Container? [online]. `http://java.dzone.com/articles/what-servlet-container`, 2013-01-05 [cit. 2014-01-02].

[6] Webové stránky serveru Jenkins CI [online]. `http://jenkins-ci.org/` .

[7] Webové stránky serveru Undertow [online]. `http://undertow.io/` .

[8] Webové stránky serveru Jetty [online]. `http://www.eclipse.org/jetty/` .

[9] Webové stránky nástroje Maven [online]. `http://maven.apache.org/what-is-maven.html` .